

# Mobile Application Development

Prof. Fernando Pinéo

# Configuração do ambiente de desenvolvimento

# Instalando o Expo



É necessário que esteja instalado o Node na máquina.

Obs: Rode o comando como administrador no powershell do Windows

```
npm install -g expo-cli
```

# Instalando o gerenciador de pacote chocolatey



O Chocolatey é um gerenciador de pacotes para o Windows. Ele facilita a instalação, atualização e remoção de programas e ferramentas de software diretamente do prompt de comando ou PowerShell, sem a necessidade de baixar manualmente os instaladores de cada programa ou configurar processos complexos.

Link: <https://chocolatey.org/install>

Executar no powershell do Windows com administrador

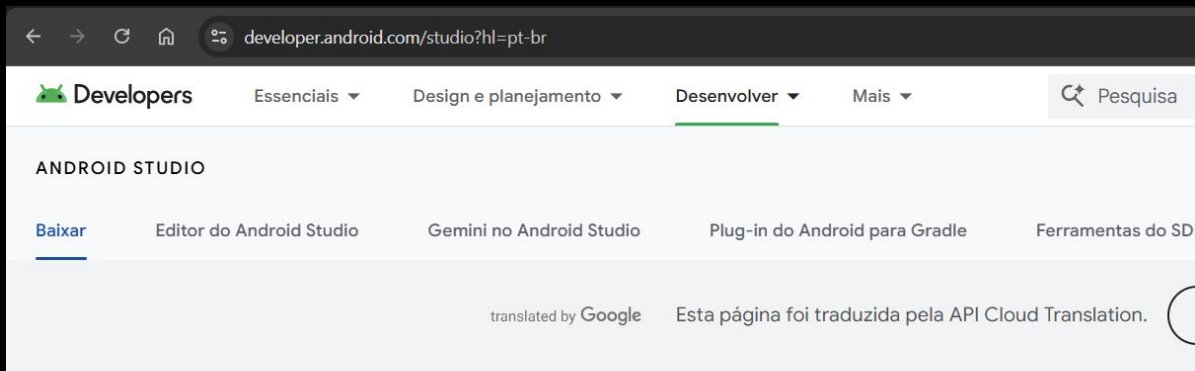
# Instalar nodejs-lts e microsoft-openjdk17

FILAP

```
choco install -y nodejs-lts microsoft-openjdk17
```


# Instalar Android Studio

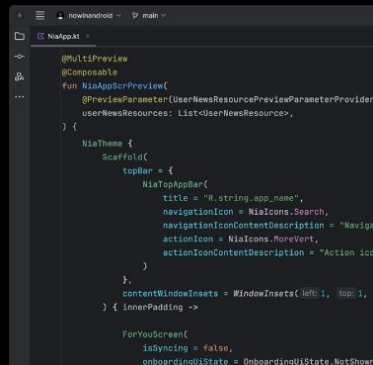
FIAP



## Android Studio

O ambiente de desenvolvimento integrado oficial para apps Android agora acelera sua produtividade com o Gemini no Android Studio, seu assistente de programação com tecnologia de IA.

Download da versão do Android Studio Ladybug 

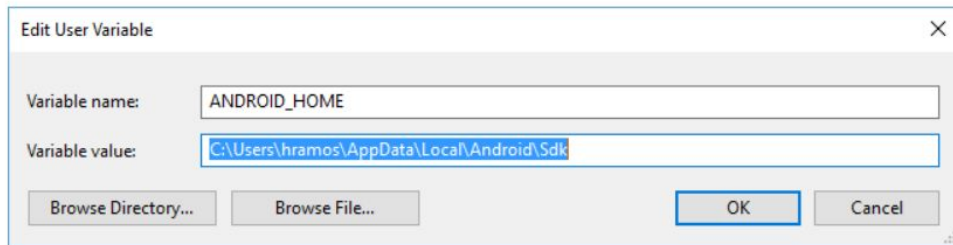


# Variável de ambiente ANDROID\_HOME FIA/P

## 3. Configure the ANDROID\_HOME environment variable

The React Native tools require some environment variables to be set up in order to build apps with n

1. Open the **Windows Control Panel**.
2. Click on **User Accounts**, then click **User Accounts** again
3. Click on **Change my environment variables**
4. Click on **New...** to create a new `ANDROID_HOME` user variable that points to the path to your Andro



Edit User Variable

Variable name:

Variable value:

The SDK is installed, by default, at the following location:

powershell

```
%LOCALAPPDATA%\Android\Sdk
```

## 4. Add platform-tools to Path

1. Open the **Windows Control Panel**.
2. Click on **User Accounts**, then click **User Accounts** again
3. Click on **Change my environment variables**
4. Select the **Path** variable.
5. Click **Edit**.
6. Click **New** and add the path to platform-tools to the list.

The default location for this folder is:

```
powershell
```

```
%LOCALAPPDATA%\Android\Sdk\platform-tools
```



# Criando primeiro projeto com Expo



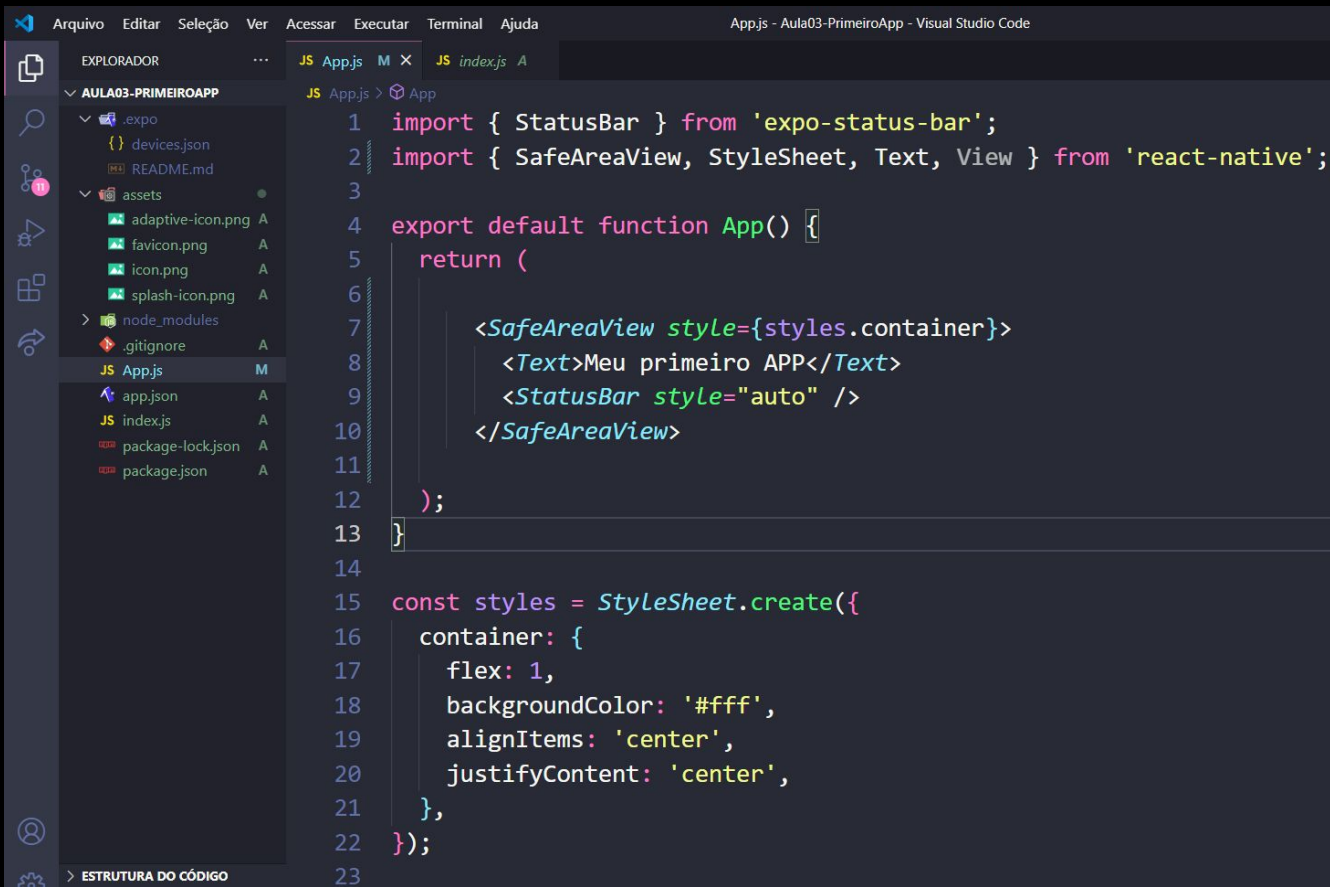
```
expo init AwesomeProject
```

```
cd AwesomeProject
```

```
npm start # you can also use: expo start
```

# Estrutura do primeiro App

FIAP

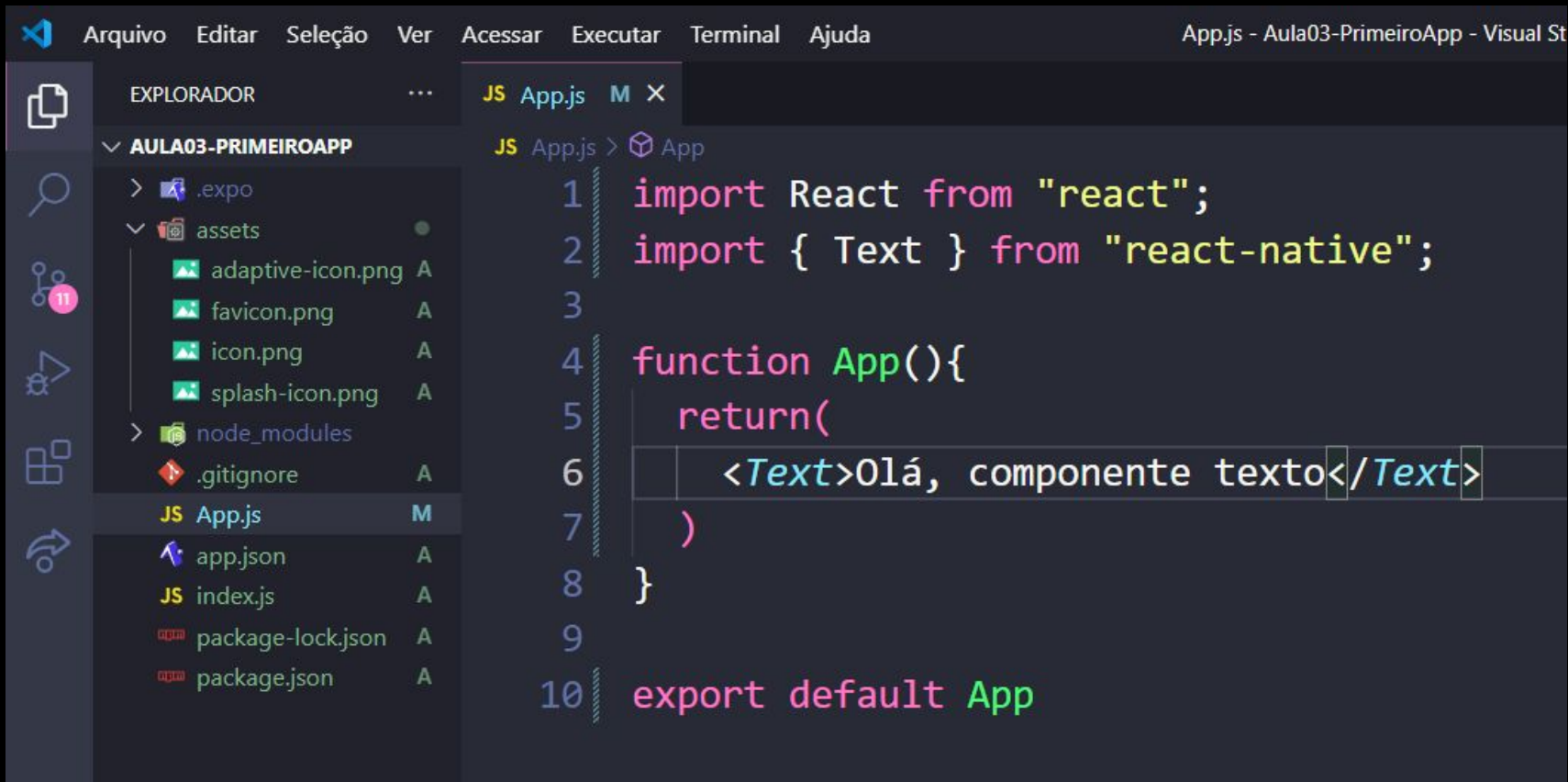


The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left and the Code Editor in the center. The Explorer sidebar displays the project structure for 'AULA03-PRIMEIROAPP', including folders like 'expo', 'assets', and 'node\_modules', and files like 'App.js', 'index.js', 'app.json', 'package-lock.json', and 'package.json'. The Code Editor shows the content of 'App.js', which is a React Native application component. The code imports 'StatusBar' from 'expo-status-bar' and 'SafeAreaView', 'StyleSheet', 'Text', and 'View' from 'react-native'. It defines a default function 'App' that returns a JSX element with a 'SafeAreaView' containing a 'Text' and a 'StatusBar'. The 'SafeAreaView' has a style object 'styles.container' and the 'Text' has the text 'Meu primeiro APP'. The 'StatusBar' has a style of 'auto'. The 'styles' object is created using 'StyleSheet.create' with a 'container' style that has a flex of 1, a background color of '#fff', and alignment and justification centered.

```
1 import { StatusBar } from 'expo-status-bar';
2 import { SafeAreaView, StyleSheet, Text, View } from 'react-native';
3
4 export default function App() {
5   return (
6     <SafeAreaView style={styles.container}>
7       <Text>Meu primeiro APP</Text>
8       <StatusBar style="auto" />
9     </SafeAreaView>
10   );
11 }
12
13
14
15 const styles = StyleSheet.create({
16   container: {
17     flex: 1,
18     backgroundColor: '#fff',
19     alignItems: 'center',
20     justifyContent: 'center',
21   },
22 });
23
```

# Componente Text

FIAP



```
Arquivo  Editor  Seleção  Ver  Acessar  Executar  Terminal  Ajuda  App.js - Aula03-PrimeiroApp - Visual St
```

EXPLORADOR

▼ AULA03-PRIMEIROAPP

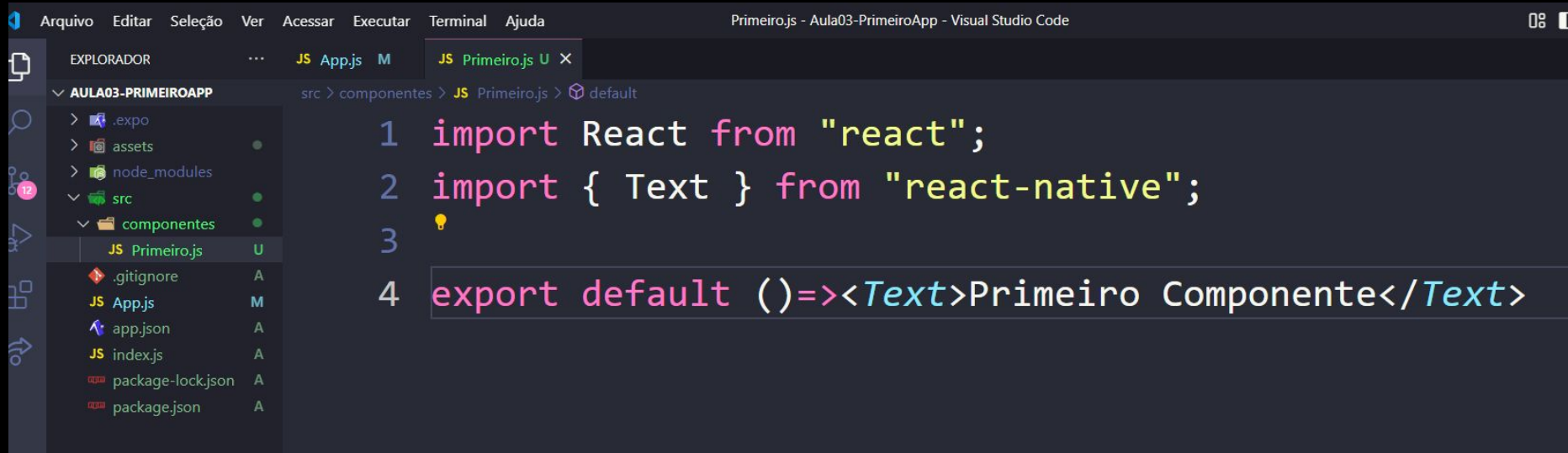
- > .expo
- ▼ assets
  - adaptive-icon.png A
  - favicon.png A
  - icon.png A
  - splash-icon.png A
- > node\_modules
- .gitignore A
- JS App.js M
- app.json A
- JS index.js A
- package-lock.json A
- package.json A

JS App.js > App

```
1  import React from "react";
2  import { Text } from "react-native";
3
4  function App(){
5    return(
6      <Text>Olá, componente texto</Text>
7    )
8  }
9
10 export default App
```

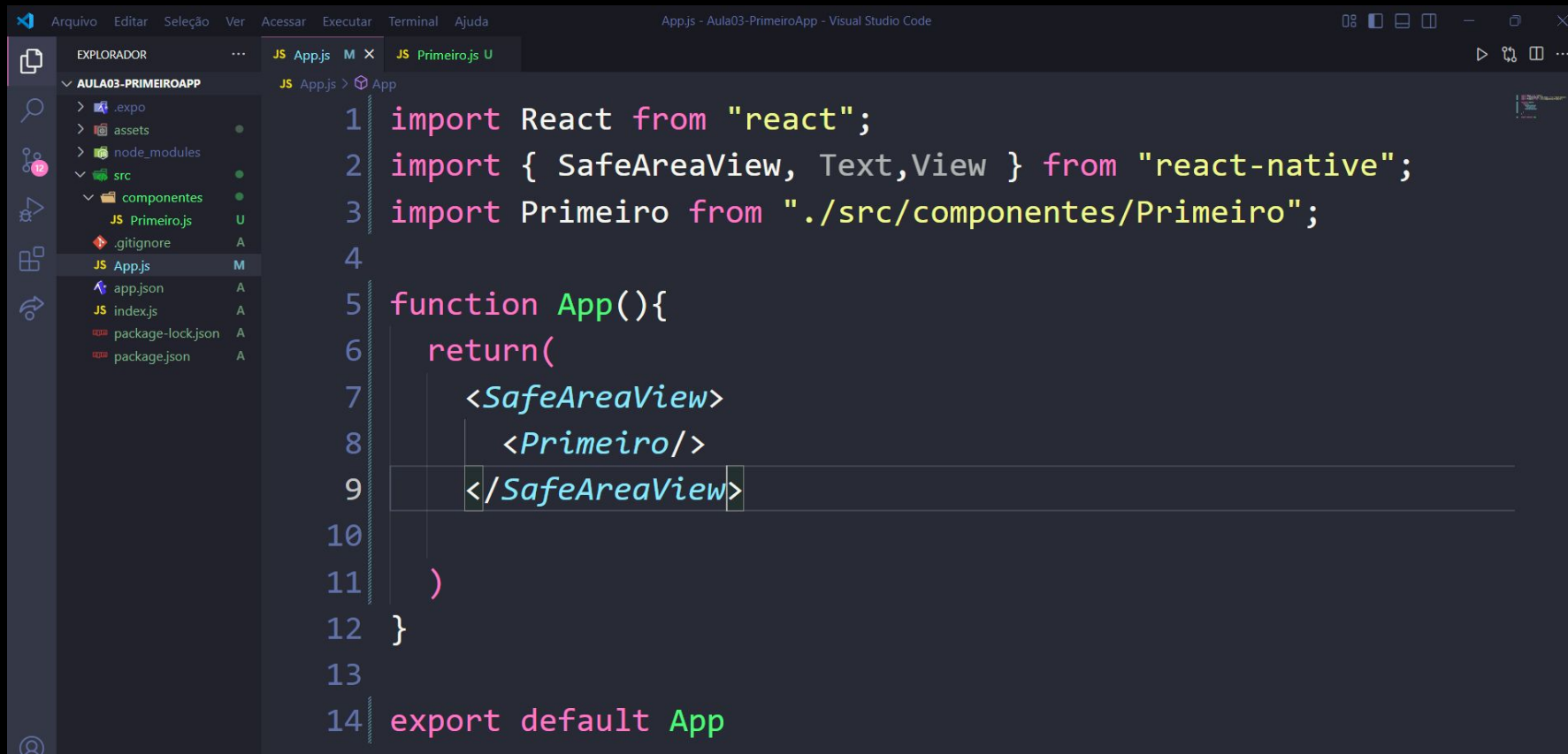
Podemos utilizar arrow function

# Criando o Primeiro componente e exportando



# Importando no App.js

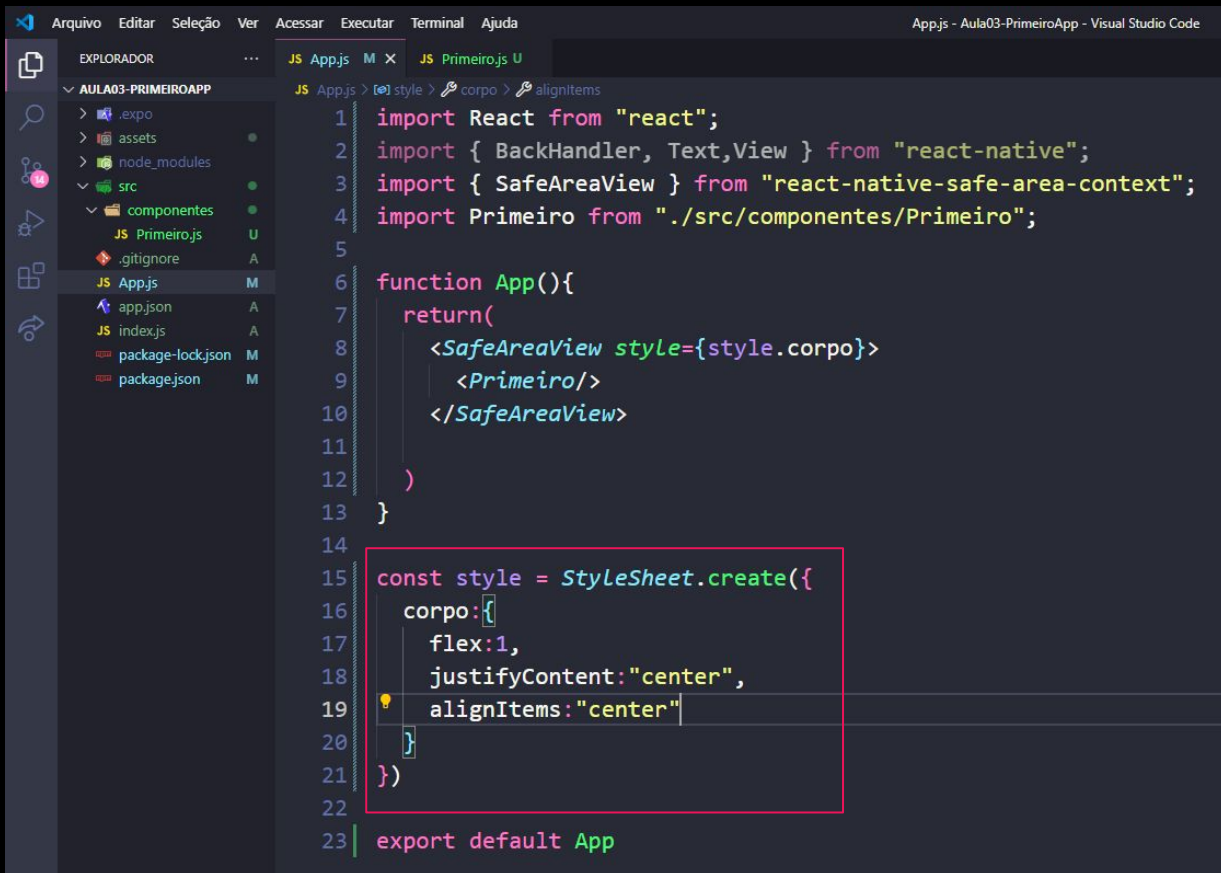
FIAP



```
1 import React from "react";
2 import { SafeAreaView, Text, View } from "react-native";
3 import Primeiro from "./src/componentes/Primeiro";
4
5 function App(){
6   return(
7     <SafeAreaView>
8       <Primeiro/>
9     </SafeAreaView>
10
11   )
12 }
13
14 export default App
```

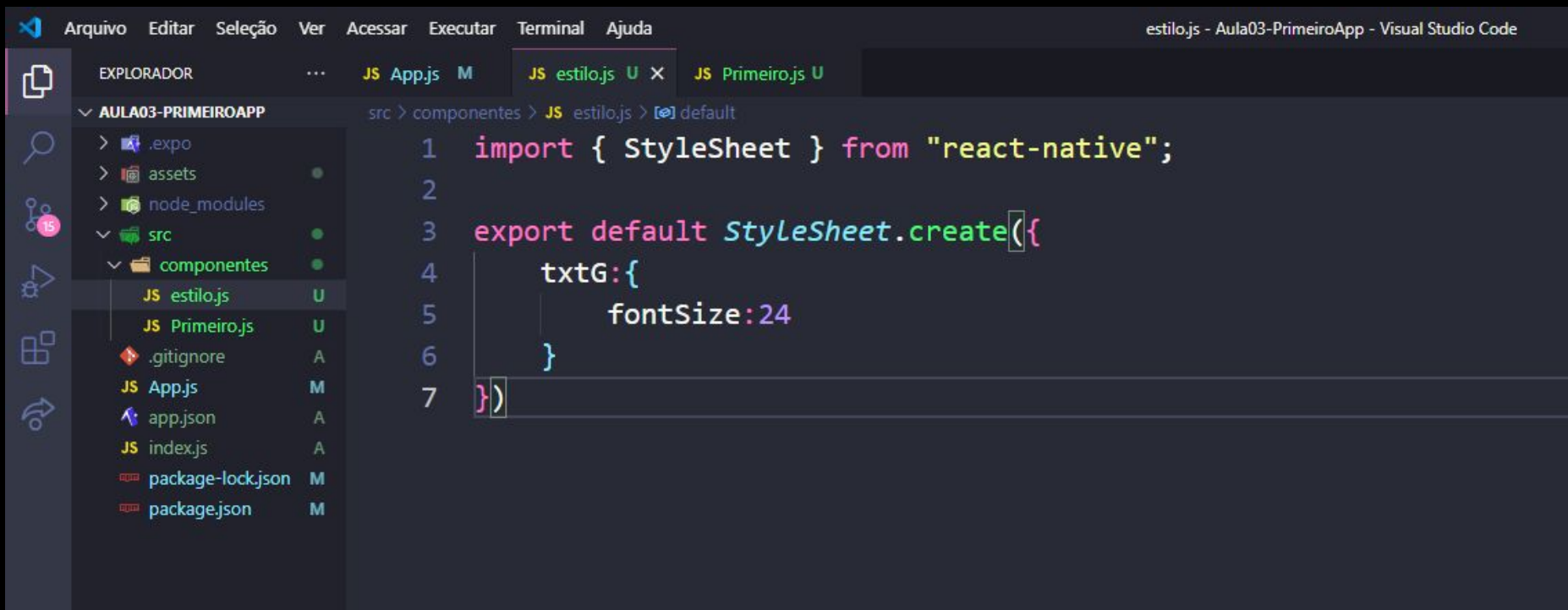
# Estilizando

FIAP



```
1 import React from "react";
2 import { BackHandler, Text, View } from "react-native";
3 import { SafeAreaView } from "react-native-safe-area-context";
4 import Primeiro from "../src/componentes/Primeiro";
5
6 function App(){
7   return(
8     <SafeAreaView style={style.corpo}>
9       <Primeiro/>
10    </SafeAreaView>
11  )
12 }
13
14
15 const style = StyleSheet.create({
16   corpo:{
17     flex:1,
18     justifyContent:"center",
19     alignItems:"center"
20   }
21 })
22
23 export default App
```

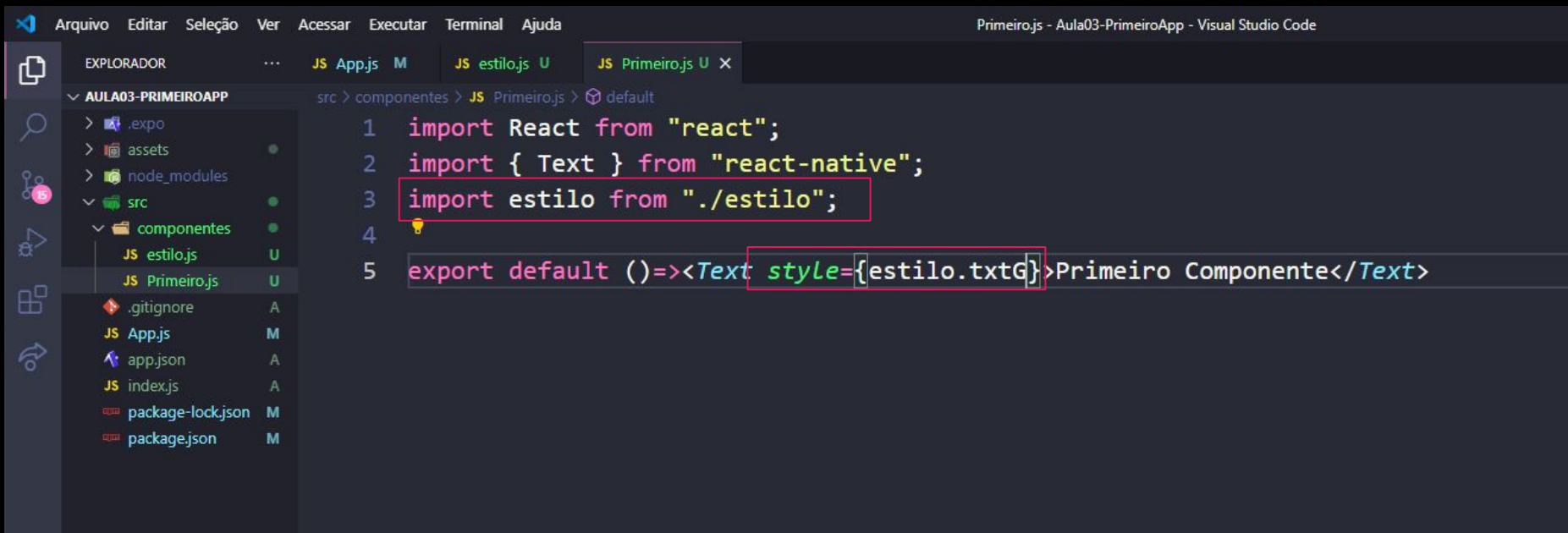
# Criando arq separado p/ o arq de estilo FIA/P





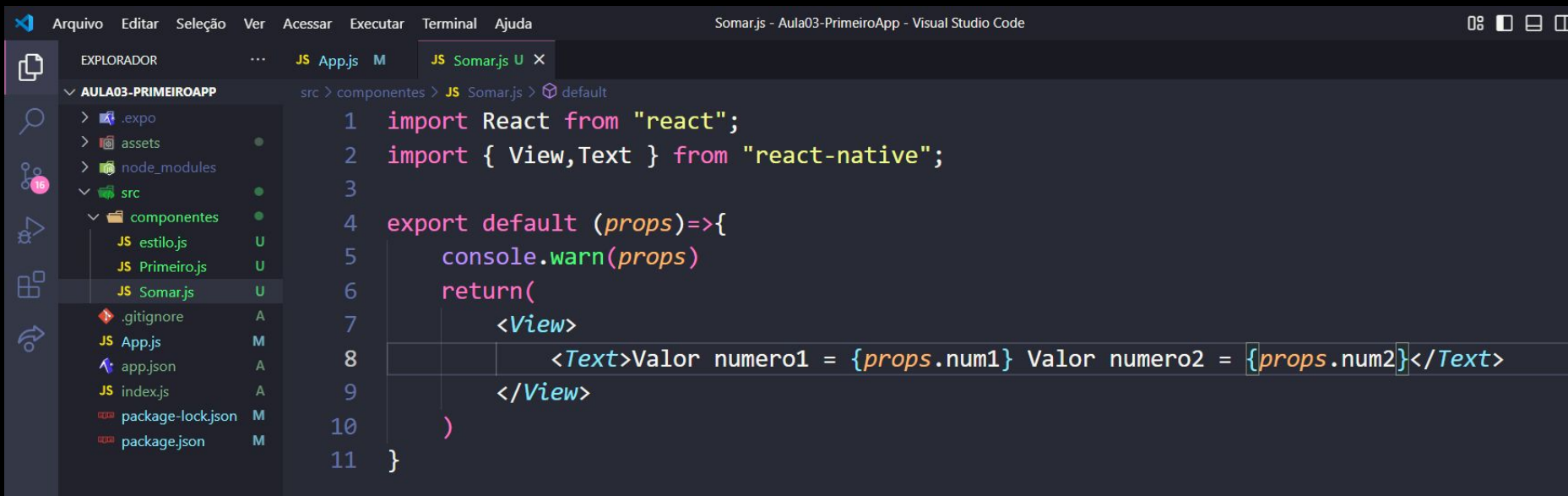
# Utilizando o estilo do arq

FIAP



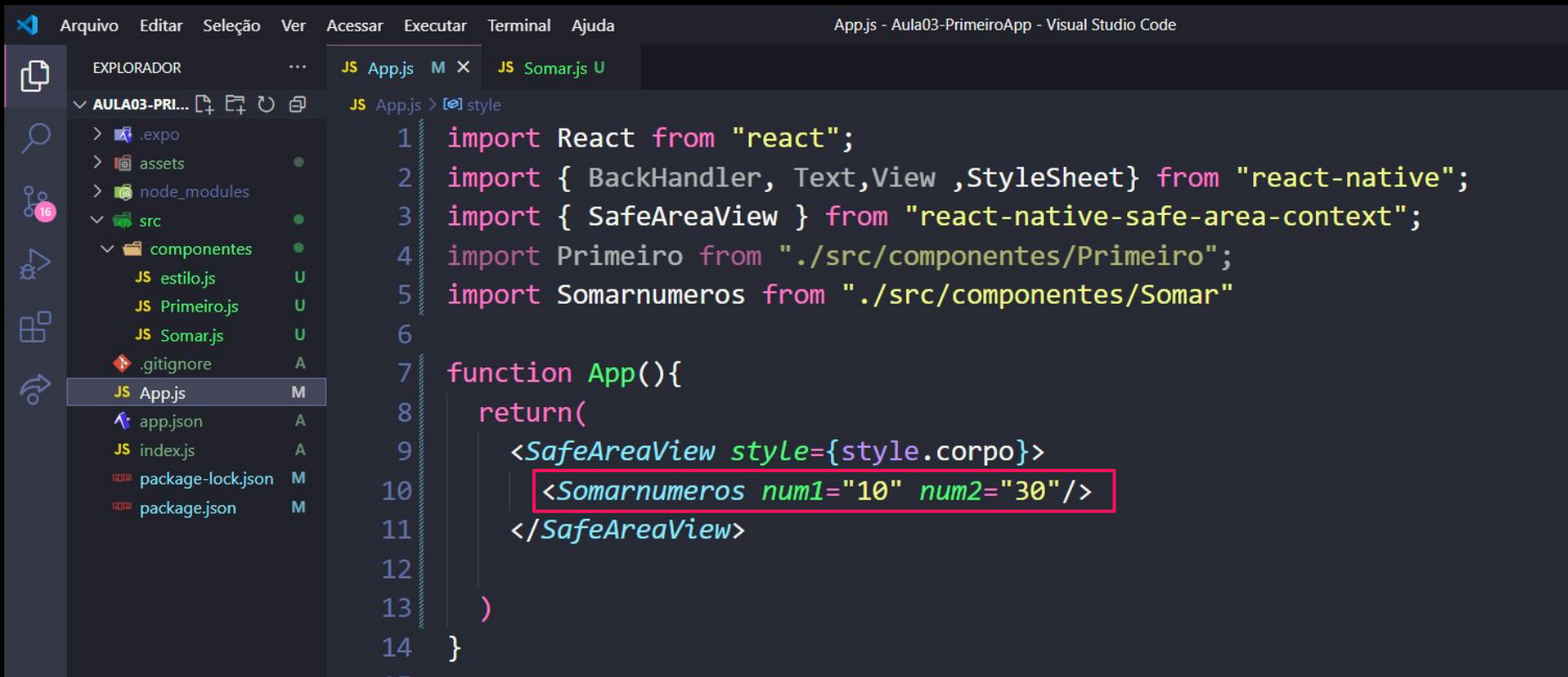
# Componente com propriedade

FIAP



# Componente com propriedades

FIAP



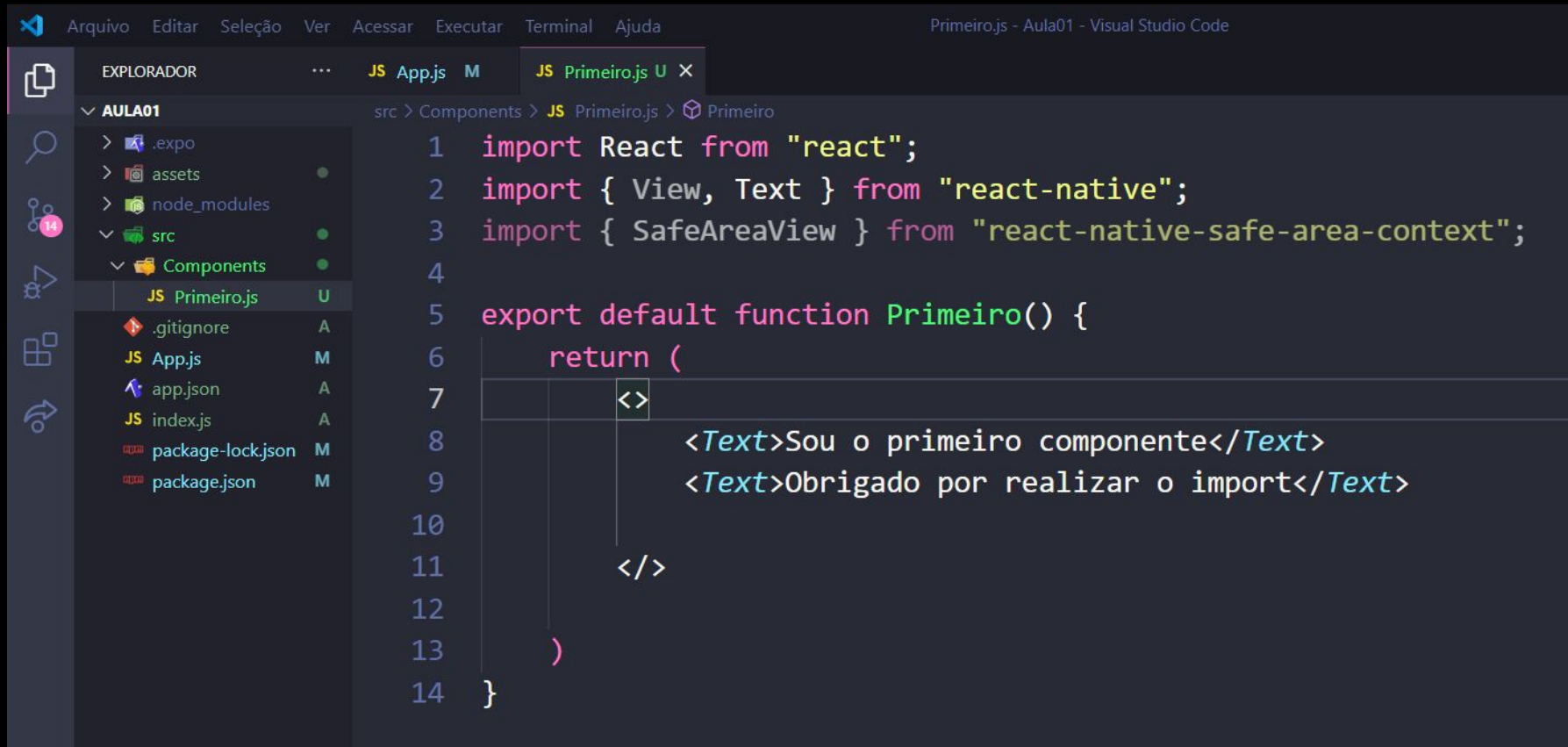
The screenshot shows the Visual Studio Code interface with a project named 'AULA03-PRIMEIROAPP'. The Explorer sidebar on the left shows the file structure, including a 'componentes' folder with files like 'estilo.js', 'Primeiro.js', and 'Somar.js'. The main editor area displays the 'App.js' file, which contains imports for 'react', 'react-native', 'react-native-safe-area-context', and local components 'Primeiro' and 'Somar'. The code defines an 'App' function that returns a JSX element. A red rectangle highlights the props 'num1="10"' and 'num2="30"' passed to the 'Somar' component within the 'SafeAreaView'.

```
1 import React from "react";
2 import { BackHandler, Text, View, StyleSheet } from "react-native";
3 import { SafeAreaView } from "react-native-safe-area-context";
4 import Primeiro from "../src/componentes/Primeiro";
5 import SomarNumeros from "../src/componentes/Somar";
6
7 function App() {
8   return (
9     <SafeAreaView style={style.corpo}>
10      <SomarNumeros num1="10" num2="30"/>
11    </SafeAreaView>
12  );
13 }
14
```

- Desenvolva no app, um componente na qual será possível realizar a passagem de 03 números como propriedades propriedades e retorne o valor maior número entre eles.
- Desenvolva um segundo componente que recebe um número como propriedade e informe se o número é positivo, negativo ou zero.

# Fragments

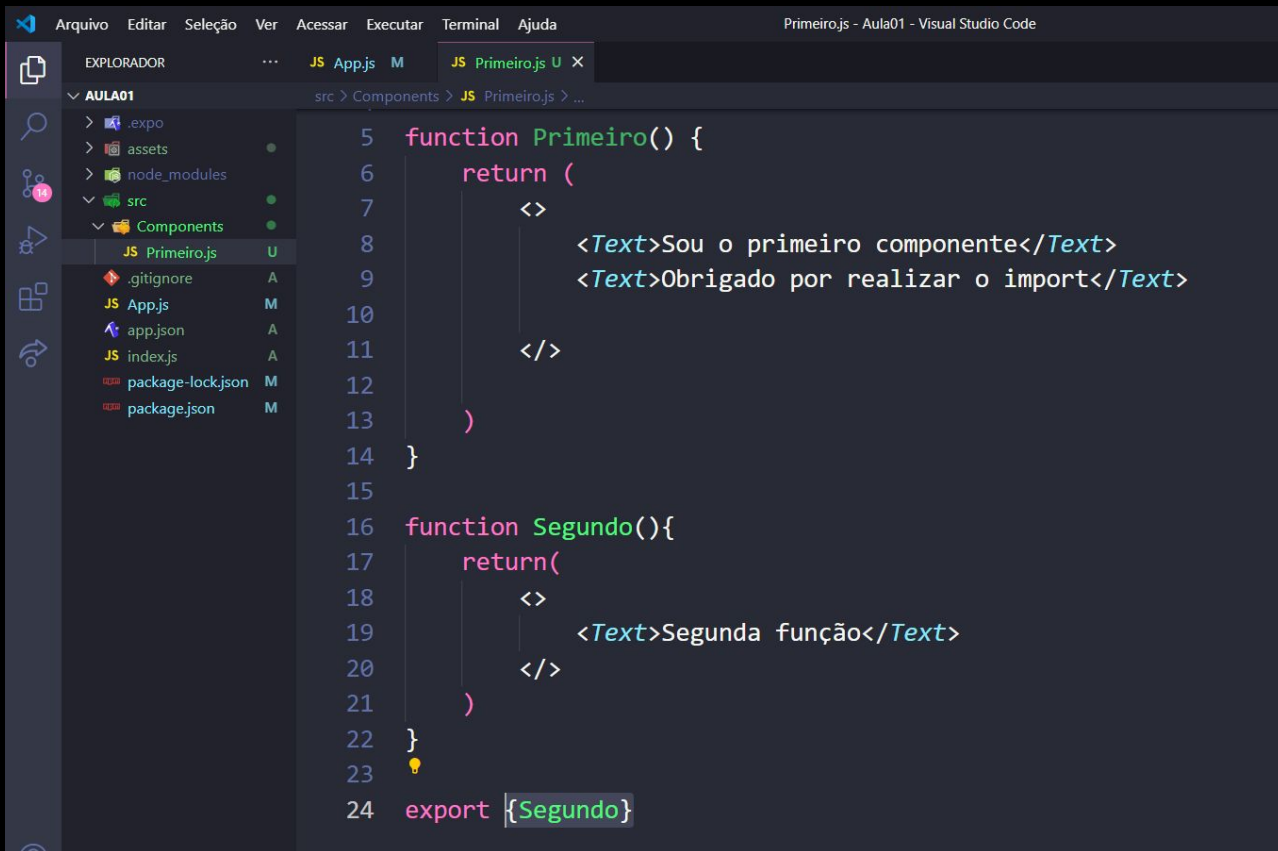
FIAP



The screenshot shows the Visual Studio Code interface with a project named 'AULA01'. The Explorer sidebar on the left displays the file structure, including folders like '.expo', 'assets', 'node\_modules', and 'src'. The 'src' folder is expanded, showing a 'Components' subfolder which contains the file 'JS Primeiro.js'. The main editor area shows the content of 'JS Primeiro.js', which is a React Native component. The code imports 'React' from 'react', 'View' and 'Text' from 'react-native', and 'SafeAreaView' from 'react-native-safe-area-context'. It then defines a default export function 'Primeiro()' which returns a JSX element containing two text components: 'Sou o primeiro componente' and 'Obrigado por realizar o import'.

```
1 import React from "react";
2 import { View, Text } from "react-native";
3 import { SafeAreaView } from "react-native-safe-area-context";
4
5 export default function Primeiro() {
6   return (
7     <>
8       <Text>Sou o primeiro componente</Text>
9       <Text>Obrigado por realizar o import</Text>
10
11     </>
12
13   )
14 }
```

# Export e Import

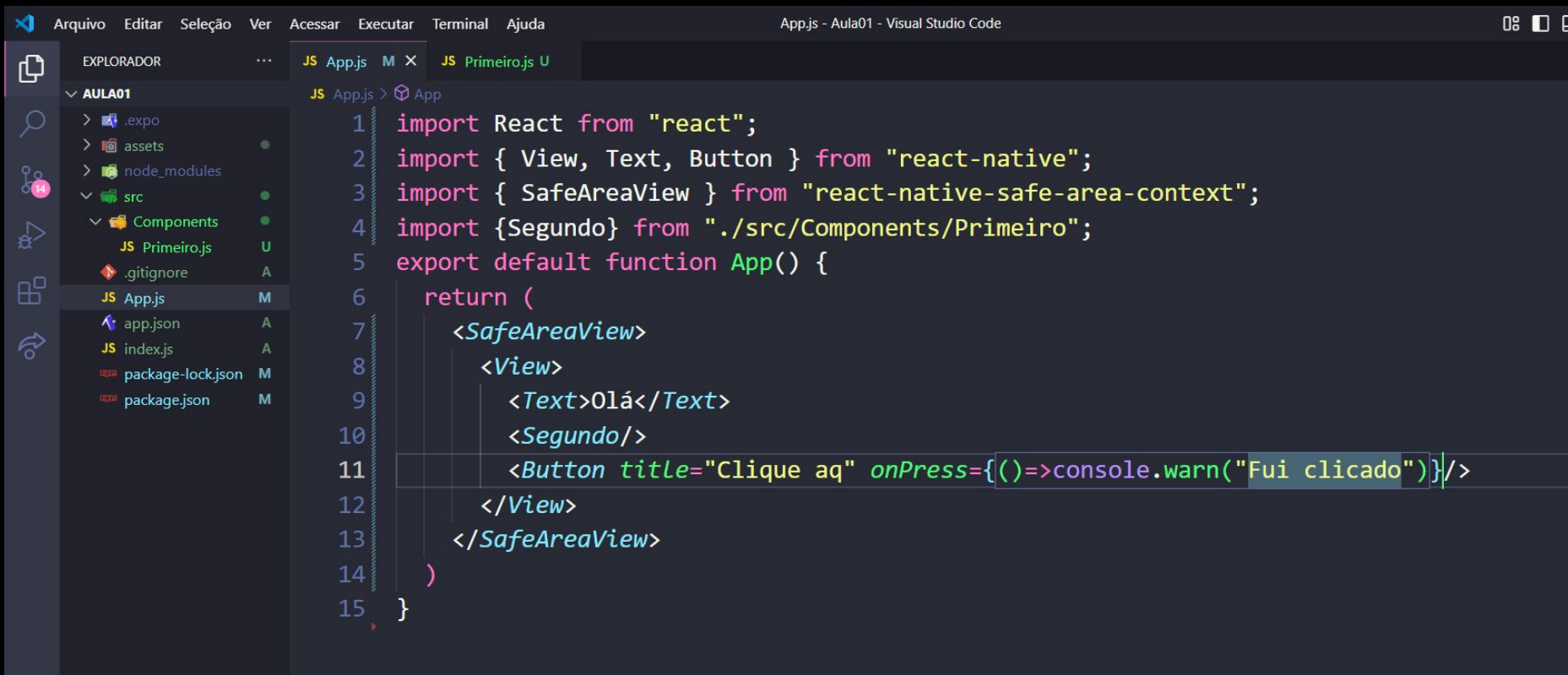


The screenshot shows the Visual Studio Code interface with a file explorer on the left and a code editor in the center. The file explorer shows a project named 'AULA01' with a 'src' directory containing a 'Components' folder. The 'Components' folder is expanded, showing 'JS Primeiro.js' selected. The code editor displays the following JavaScript code:

```
5 function Primeiro() {
6     return (
7         <>
8         <Text>Sou o primeiro componente</Text>
9         <Text>Obrigado por realizar o import</Text>
10    )
11    </>
12  }
13
14
15
16 function Segundo(){
17     return(
18         <>
19         <Text>Segunda função</Text>
20     )
21 }
22
23
24 export {Segundo}
```

# Adicionando o botão

FIAP



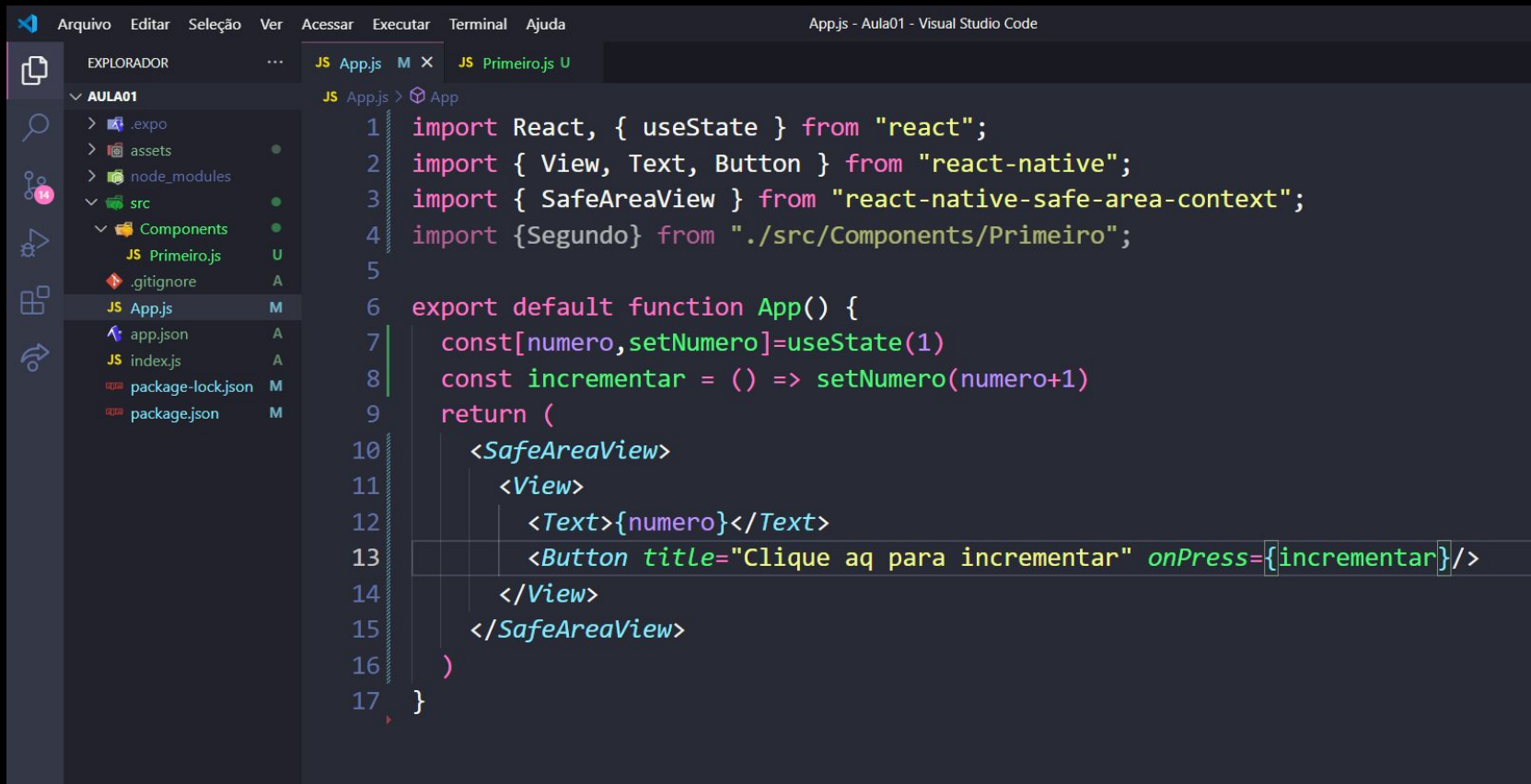
The screenshot shows the Visual Studio Code interface with a project named 'AULA01'. The Explorer sidebar on the left shows the file structure: a root directory with '.expo', 'assets', 'node\_modules', 'src', and 'Components'. Inside 'src', there is a 'Components' folder containing 'JS Primeiro.js' and a file named 'App.js'. The main editor area displays the code for 'App.js', which imports React, react-native, and react-native-safe-area-context. It also imports a 'Segundo' component from './src/Components/Primeiro'. The App function returns a JSX element with a SafeAreaView, a View containing a Text and a Segundo component, and a Button with the text 'Clique aq' and an onPress handler that logs a warning to the console.

```
1 import React from "react";
2 import { View, Text, Button } from "react-native";
3 import { SafeAreaView } from "react-native-safe-area-context";
4 import { Segundo } from "../src/Components/Primeiro";
5 export default function App() {
6   return (
7     <SafeAreaView>
8       <View>
9         <Text>Olá</Text>
10        <Segundo/>
11        <Button title="Clique aq" onPress={()=>console.warn("Fui clicado")}/>
12      </View>
13    </SafeAreaView>
14  )
15 }
```



# useState

FIAP



The image shows a screenshot of the Visual Studio Code editor interface. The top bar displays the menu (Arquivo, Editar, Seleção, Ver, Acessar, Executar, Terminal, Ajuda) and the title bar (App.js - Aula01 - Visual Studio Code). The left sidebar shows the Explorer (EXPLORADOR) with a file tree for 'AULA01' containing folders like .expo, assets, node\_modules, and src, and files like app.json, index.js, package-lock.json, and package.json. The main editor area shows the code for 'App.js' with the following content:

```
1 import React, { useState } from "react";
2 import { View, Text, Button } from "react-native";
3 import { SafeAreaView } from "react-native-safe-area-context";
4 import { Segundo } from "../src/Components/Primeiro";
5
6 export default function App() {
7   const [numero, setNumero] = useState(1)
8   const incrementar = () => setNumero(numero+1)
9   return (
10     <SafeAreaView>
11       <View>
12         <Text>{numero}</Text>
13         <Button title="Clique aq para incrementar" onPress={incrementar}/>
14       </View>
15     </SafeAreaView>
16   )
17 }
```



Dúvidas?